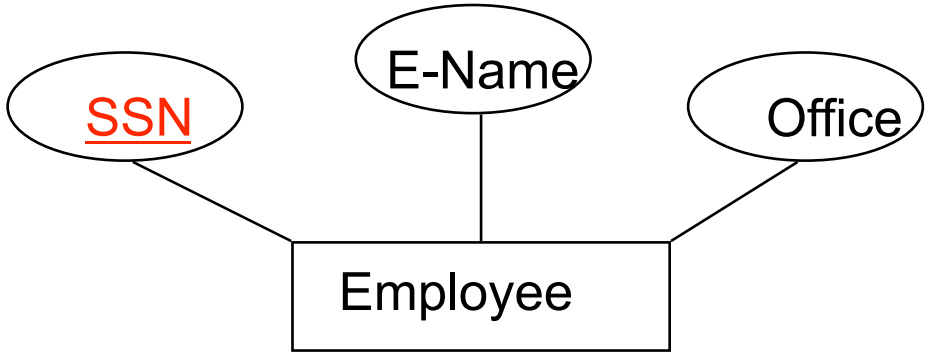
Translating an ER Diagram to a Relational Schema

Translate each entity set into a table, with keys.

Entity set:

- represented as a table
- has a key … which becomes a key for the table
- All single-valued attributes become attributes of the table
- Create a new table for any multi-valued attributes

If employee has many offices – add an extra table.




```
CREATE TABLE Employee
        (SSN CHAR(11),
        E-Name CHAR(20),
        Office INTEGER,
        PRIMARY KEY (SSN))
```

Or delete Office from above and add:

```
CREATE TABLE Emp-Office
        (SSN CHAR(11),
        Office INTEGER,
        PRIMARY KEY (SSN, Office))
```

Sample Data

Project (P-number, P-name, Due-Date)

Employee (SSN, E-Name, **Office**)

12	Smith	O-105
15	Wei	O-110
20	Jones	O-112

Notice the key for the new table: Office-Assignment .

Project (P-number, P-name, Due-Date)

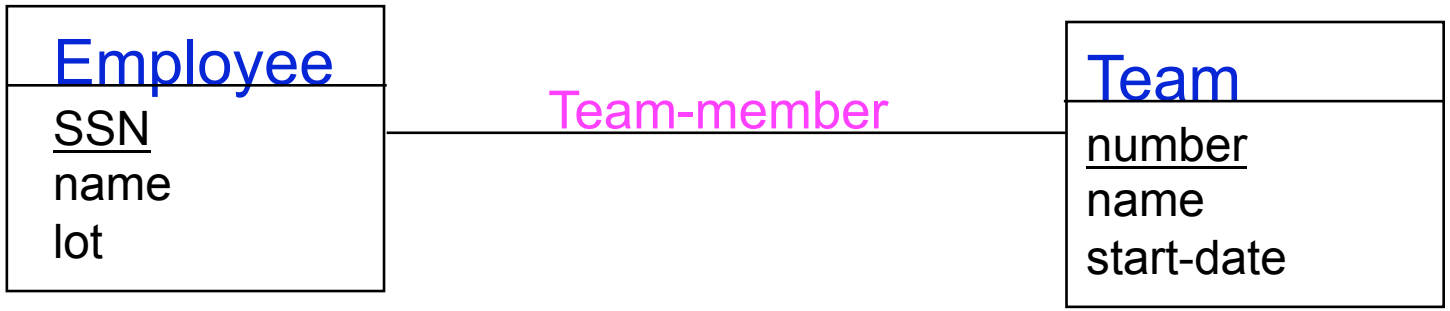
Employee (SSN, E-Name)

12	Smith
15	Wei

Office-Assignment (SSN, Office)

12	O-105
12	O-106
15	O-110

What data do we need to record a relationship?



We must indicate which employee and which team we want to have connected (for each relationship).

We need the key value for an employee and the key value for the team – stored together – to represent the relationship. Could be in an existing table that represents one of the involved entities or a new table (introduced to represent the relationship).

We may want additional attributes – about the relationship.

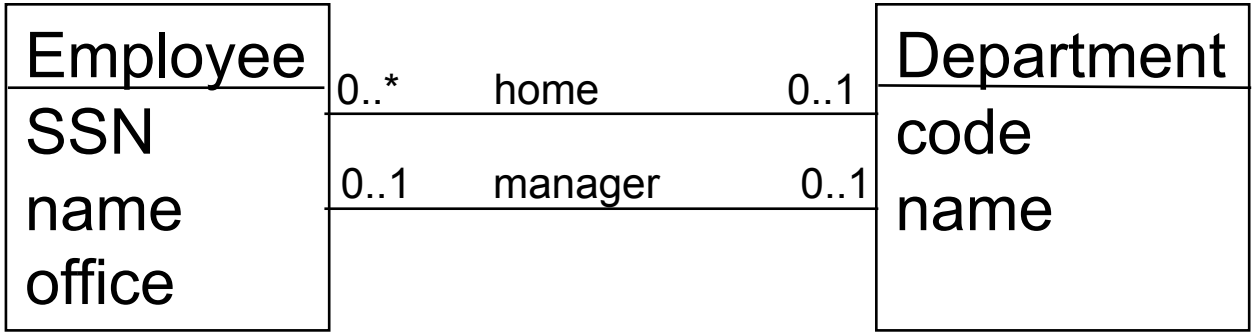
We need: Team-rel (SSN, number)

Translate each **many-to-many** relationship set into a new table.

Team-member (SSN, number) with 2 foreign keys

Team (number, name, start-date)
Employee (SSN, name, lot)

Translate one-to-many relationship set into a foreign key. (Or a separate table.)

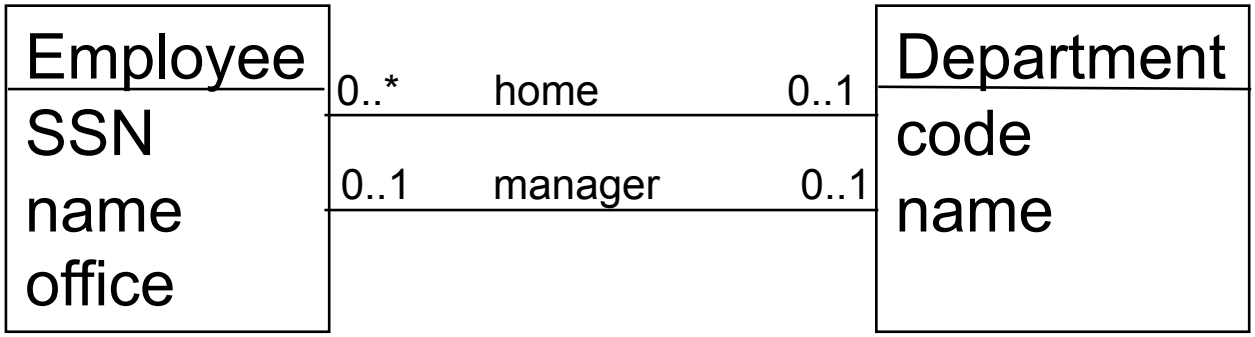


- Add a foreign key to the "many" side of the relationship to represent a 1-to-many relationship set.

Department (<u>code</u>, name)
Employee (<u>SSN</u>, name, office, dept)
dept is a foreign key (referencing Department.code)

Notice: SSN and code are side by side, in the same table, to represent the home relationship set.

Or create a table for 1-many relationship



- Department (code, name)
  Employee (SSN, name, office, dept)

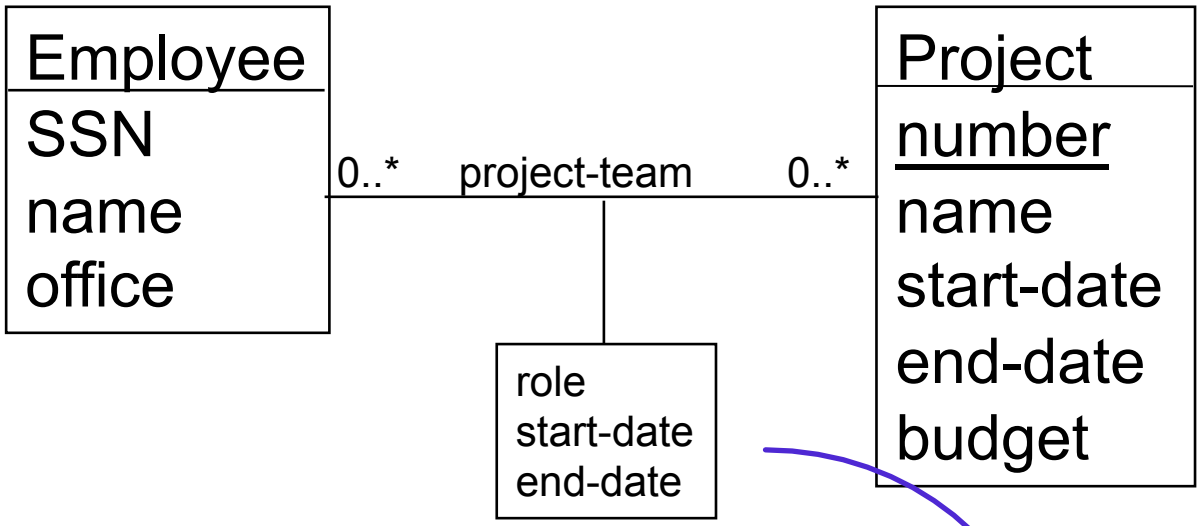
vs.

Or...Create a newtable for a 1-many relationship set.

Home-dept(SSN, code)
Department (code, name)
Employee (SSN, name, office)

What are the tradeoffs between these two?

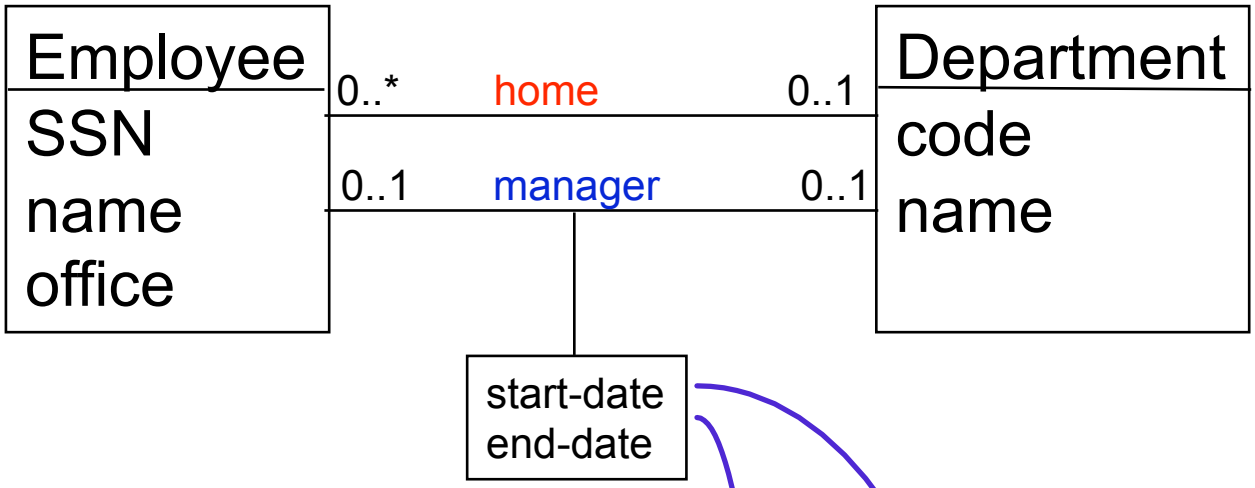
Note:
SSN is the key for Home-dept



When a many-many relationship set has attributes:

Project-team (number, SSN, role, start-date, end-date)
Project (number, name, start-date, end-date, budget)
Employee (SSN, name, office)

Put them in the table that represents the relationship.



When a 1-to-many relationship set has an attribute: add them to the table where the relationship is represented.

Department (code, name, manager, start-date, end-date)
Employee (SSN, name, office, dept)
or
Manager (SSN, code, start-date, end-date)

Participation Constraints in SQL

- We can require any table to be in a binary relationship using a foreign key which is required to be NOT NULL
  (but little else without resorting to CHECK constraints)

```
CREATE TABLE  Department (
   code                  INTEGER,
   name                  CHAR(20),
   manager               CHAR(9) NOT NULL,
   start-date            DATE,
   end-date              DATE,
   PRIMARY KEY  (d-code),
   FOREIGN KEY  (manager-ssn) REFERENCES Employee,
      ON DELETE NO ACTION)
```

Translating Weak Entity Sets

- Weak entity sets and identifying relationship sets are translated into a single table. Must include key of strong entity set, as a foreign key.
- Key for this table is key of strong entity + partial key.
- When the owner entity is deleted, all owned weak entities must also be deleted.

```
CREATE TABLE  Insurance_Policy (
   dep-name        CHAR(20),
   age             INTEGER,
   cost            REAL,
   ssn             CHAR(11) NOT NULL,
   PRIMARY KEY  (dep-name, ssn),
   FOREIGN KEY  (ssn) REFERENCES Employee,
     ON DELETE CASCADE)
```

Summary of Translation Steps: ER to Tables

1. Create table and choose key for each entity set; include single-valued attributes.
2. Create table for each weak entity set; include single-valued attributes. Include key of owner as a foreign key in the weak entity. Set key as foreign key of owner plus local, partial key.
3. For each 1:1 relationship set, add a foreign key to one of the entity sets involved in the relationship (a foreign key to the other entity in the relationship)*.
4. For each 1:N relationship set, add a foreign key to the entity set on the N-side of the relationship (to reference the entity set on the 1-side of the relationship)*.
5. For each M:N relationship set, create a new table. Include a foreign key for each participant entity set, in the relationship set. The key for the new table is the set of all such foreign keys.
6. For each multi-valued attribute, construct a separate table. Repeat the key for the entity in this new table. It will serve as both the key for this table as well as a foreign key to the original table for the entity.

* Unless relationship set has attributes. If it does, create a new table for the relationship set.

This algorithm from Elmasri/Navathe, p. 174.